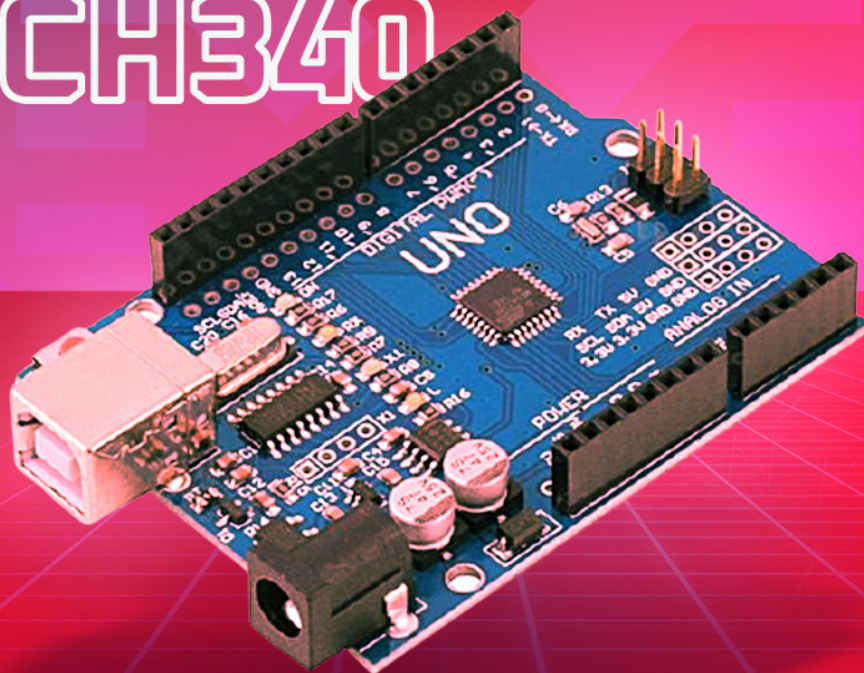
APOSTILA KIT

ARDUINO INICIANTE

Especial

PLACA UNO

CH340

WWW.ELETROGATE.COM

V1.5 – Dezembro/2023

Olá, **Maker**!

Primeiramente, gostaríamos de parabenizá-lo(a) pelo interesse neste material.

Esta apostila exclusiva servirá como material de apoio teórico e prático para o Kit Arduino Iniciante, mas, mesmo que você ainda não tenha adquirido o seu Kit, ela poderá te ajudar a conhecer um pouco mais sobre o Universo Maker, de uma maneira didática e objetiva.

Caso você nunca tenha ouvido falar sobre o Arduino, não se preocupe. Esta apostila também é para você! Além de uma introdução sobre esta plataforma, instalação e configuração, você também aprenderá como utilizar todos os recursos que a sua placa Arduino oferece, através de projetos práticos, desde a programação, montagem, até o projeto em funcionamento.

Vale ressaltar que, embora esta apostila proponha alguns exemplos de projetos práticos, utilizando os componentes inclusos no Kit, existem infinitas possibilidades para você criar diversos projetos, utilizando componentes como: sensores, motores e shields, dentre outros.

Caso ainda não tenha adquirido seu Kit, acesse o nosso site e garanta o seu.

Desejamos bons estudos!

Sumário

Parte I - Revisão de circuitos elétricos e componentes básicos

Introdução

A primeira parte da apostila faz uma revisão sobre circuitos elétricos, com destaque para questões práticas de montagem e segurança que surgem no dia a dia do usuário do Kit Arduino Iniciante.

O conteúdo de circuitos elétricos aborda divisores de tensão e corrente, conversão de níveis lógicos, grandezas analógicas e digitais, níveis de tensão e cuidados práticos de montagem. Em relação aos componentes básicos, é feita uma breve discussão sobre resistores elétricos, capacitores, leds, diodos, chaves e protoboards.

A Arduino Uno é o principal componente do Kit e é discutida e introduzida em uma seção a parte, na Parte II da apostila.

Todos os conteúdos da Parte I são focados na apostila Arduino Iniciante, tendo em vista a utilização adequada dos componentes e da realização prática de montagens pelos usuários. No entanto, recomenda-se a leitura das referências indicadas ao final de cada seção para maior aprofundamento.

O leitor veterano, já acostumado e conhecedor dos conceitos essenciais de eletrônica e eletricidade, pode pular a Parte I e ir direto a Parte II, na qual são apresentadas uma seção de exemplo de montagem para cada sensor ou componente importante da apostila. Algum conhecimento prévio é bem-vindo, mas não é necessário para utilização do kit. Caso seja seu primeiro contato, nós recomendamos que antes de fazer as montagens você leia esses três artigos do http://blog.eletrogate.com/

- https://blog.eletrogate.com/o-que-e-arduino-para-que-serve-vantagens-e-como-utilizar/
- *http://blog.eletrogate.com/programacao-arduino-parte-1/*
- *http://blog.eletrogate.com/programacao-arduino-parte-2/*

Preparado? Vamos começar!

Revisão de circuitos elétricos

A apostila Arduino Iniciante, bem como todas as outras apostilas que tratam de Arduino e eletrônica em geral, tem como conhecimento de base as teorias de circuitos elétricos e de eletrônica analógica e digital.

Do ponto de vista da teoria de circuitos elétricos, é importante conhecer os conceitos de grandezas elétricas: Tensão, corrente, carga, energia potência elétrica. Em todos os textos sobre Arduino ou qualquer assunto que envolva eletrônica, você sempre terá que lidar com esses termos. Para o leitor que se inicia nessa seara, recomendamos desde já que mesmo que a eletrônica não seja sua área de formação, que conheça esses conceitos básicos.

Vamos começar pela definição de “circuito elétrico”. ***Um circuito elétrico/eletrônico é uma interconexão de elementos elétricos/eletrônicos***. Essa interconexão pode ser feita para atender a uma determinada tarefa, como acender uma lâmpada, acionar um motor, dissipar calor em uma resistência e tantos outras.

O circuito pode estar **energizado** ou **desenergizado**. Quando está energizado, é quando uma fonte de tensão externa ou interna está ligada aos componentes do circuito. Nesse caso, uma corrente elétrica fluirá entre os condutores do circuito. Quando está desenergizado, a fonte de tensão não está conectada e não há corrente elétrica fluindo entre os condutores.

Mas atenção, alguns elementos básicos de circuitos, como os capacitores ou massas metálicas, são elementos que armazenam energia elétrica. Em alguns casos, mesmo não havendo fonte de tensão conectada a um circuito, pode ser que um elemento que tenha energia armazenada descarregue essa energia dando origem a uma corrente elétrica transitória no circuito. Evitar que elementos do circuito fiquem energizados mesmo sem uma fonte de tensão, o que pode provocar descargas elétricas posteriores (e em alguns casos, danificar o circuito ou causar choques elétricos) é um dos motivos dos sistemas de aterramento em equipamentos como osciloscópios e em instalações residenciais, por exemplo.

Em todo circuito você vai ouvir falar das grandezas elétricas principais, assim, vamos aprender o que é cada uma delas.

Carga e corrente elétrica

A grandeza mais básica nos circuitos elétricos é a carga elétrica. Carga é a propriedade elétrica das partículas atômicas que compõem a matéria, e é medida em Coulombs. Sabemos da física elementar que a matéria é constituída de elétrons, prótons e neutros. **A carga elementar é a carga de 1 elétron, que é igual a 1,602 x10-19 C**.

Do conceito de carga elétrica advém o **conceito de corrente elétrica, que nada mais é do que a taxa de variação da carga em relação ao tempo**, ou seja, quando você tem um fluxo de carga em um condutor, a quantidade de carga (Coulomb) que atravessa esse condutor por unidade de tempo, é chamada de corrente elétrica. A medida utilizada para corrente é o **Ampére(A).**

Aqui temos que fazer uma distinção importante. Existem corrente elétrica contínua e alternada:

- **Corrente elétrica contínua:** É uma corrente que permanece constante e em uma única direção durante todo o tempo.
- **Corrente elétrica alternada**: É uma corrente que varia senoidalmente (ou de outra forma) com o tempo.

Com o Arduino UNO, lidamos como correntes elétricas contínuas, pois elas fluem sempre em uma mesma direção. É diferente da corrente e tensão elétrica da tomada de sua casa, que são alternadas.

Ou seja, os seus circuitos com Arduino UNO sempre serão alimentados com grandezas contínuas (corrente e tensão contínuas).

Tensão elétrica

Para que haja corrente elétrica em um condutor, é preciso que os elétrons se movimentem por ele em uma determinada direção, ou seja, é necessário "alguém" para transferir energia para as cargas elétricas para movê-las. Isso é feito por uma força chamada **força eletromotriz (fem),** tipicamente representada por uma bateria. Outros dois nomes comuns para força eletromotriz são **tensão elétrica** e **diferença de potencial.** O mais comum é você ver apenas "*tensão*" nos artigos e exemplos com Arduino. Assim, definindo formalmente o conceito: Tensão elétrica é a energia necessária para mover uma unidade de carga através de um elemento, e é medida em **Volts (V).**

Potência e energia

A tensão e a corrente elétrica são duas grandezas básicas, e juntamente com a potência e energia, são as grandezas que descrevem qualquer circuito elétrico ou eletrônico. A potência é definida como a variação de energia (que pode estar sendo liberada ou consumida) em função do tempo, e é medida em **Watts (W).** A potência está associada ao calor que um componente está dissipando e a energia que ele consume.

Nós sabemos da vida prática que uma lâmpada de 100W consome mais energia do que uma de 60 W. Ou seja, se ambas estiverem ligadas por 1 hora por exemplo, a lâmpada de 100W vai implicar numa conta de energia mais cara.
A potência se relaciona com a tensão e corrente pela seguinte equação:

$$P = V \times I$$

Essa é a chamada potência instantânea. Com essa equação você saber qual a potência dissipada em um resistor por exemplo, bastando que saiba a tensão nos terminais do resistor e a corrente que flui por ele. O conceito de potência é importante pois muitos hobbystas acabam não tendo noção de quais valores de resistência usar, ou mesmo saber especificar componentes de forma adequada.

Um resistor de 33 ohms de potência igual a 1/4W, por exemplo, não pode ser ligado diretamente em circuito de 5V, pois nesse caso a potência dissipada nele seria maior que a que ele pode suportar.

Vamos voltar a esse assunto em breve, por ora, tenha em mente que é importante ter uma noção da potência dissipada ou consumida pelos elementos de um circuito que você vá montar.

Por fim, a energia elétrica é o somatório da potência elétrica durante todo o tempo em que o circuito esteve em funcionamento. A energia é dada em **Joules (J)** ou **Wh (watt-hora).** A unidade Wh é interessante pois mostra que a energia é calculada multiplicando-se a potência pelo tempo(apenas para os casos em que a potência é constante).

Essa primeira parte é um pouco conceitual, mas é importante saber de onde vieram toda a terminologia que você sempre vai ler nos manuais e artigos na internet. Na próxima seção, vamos discutir os componentes básicos de circuito que compõem o Kit Arduino Iniciante.

Conversão de níveis lógicos e alimentação correta de circuitos

É muito comum que hobbystas e projetistas em geral acabem por cometer alguns erros de vez em quando. Na verdade, mesmo alguns artigos na internet e montagens amplamente usadas muitas vezes acabam por não utilizar as melhores práticas de forma rigorosa. Isso acontece muito no caso dos níveis lógicos dos sinais usados para interfacear o Arduino com outros circuitos.

Como veremos na seção de apresentação da Arduino Uno, ela é alimentada por um cabo USB ou uma fonte externa entre 6V e 12V. O Circuito da Arduino Uno possui reguladores de tensão que convertem a alimentação de entrada para 5V e para 3V. Os sinais digitais das portas de saída (I/Os) da Arduino variam entre 0 e 5V.

Isso significa que quando você quiser usar o seu Arduino UNO com um sensor ou CI que trabalhe com 3.3V, é preciso fazer a adequação dos níveis de tensão, pois se você enviar um sinal de 5V (saída do Arduino) em um circuito de 3.3V (CI ou sensor), você poderá queimar o pino daquele componente.

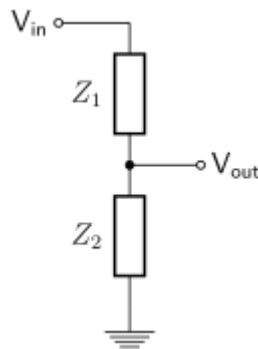
Em geral, sempre que dois circuitos que trabalhem com níveis de tensão diferentes forem conectados, é preciso fazer a conversão dos níveis lógicos. O mais comum é ter que abaixar saídas de 5V para 3.3V. Subir os sinais de 3.3V para 5V na maioria das vezes não é necessário pois o Arduino entende 3.3V como nível lógico alto, isto é, equivalente a 5V.

Para fazer a conversão de níveis lógicos você tem duas opções:

- Usar um divisor de tensão;
- Usar um CI conversor de níveis lógicos;

O divisor de tensão é a solução mais simples, mas usar um CI conversor é mais elegante e é o ideal. O divisor de tensão consiste em dois resistores ligados em série (Z1 e Z2), em que o sinal de 5V é aplicado a o terminal de um deles. O terminal do segundo resistor é ligado ao GND, e o ponto de conexão entre os dois resistores é a saída do divisor, cuja tensão é dada pela seguinte relação:

$$V_{\mathrm{out}} = \frac{Z_2}{Z_1 + Z_2} \cdot V_{\mathrm{in}}$$

Em que Z1 e Z2 são os valores dos resistores da figura a seguir.

Um divisor de tensão muito comum é fazer Z1 igual 330 ohms e Z2 igual 680 ohms. Dessa forma a saída Vout fica sendo 3.336 V. Como no Kit Arduino Iniciante não há resistor de 680ohms, você pode ligar um de 330ohms como Z1 e dois de 330ohms como Z2. Os dois de 330 ligados em série formam um resistor de 660 ohm, o que resulta numa saída de 3.33V.

Vamos exemplificar como fazer um divisor de tensão como esse na seção de exemplos da parte II da apostila.

Revisão de componentes eletrônicos

O Kit Arduino Iniciante possui os seguintes componentes básicos para montagens de circuitos:

- Buzzer Ativo 5V,
- LED Vermelho/ Verde/ Amarelo,
- Resistor 330Ω/ 1KΩ/ 10KΩ,
- Diodo 1N4007,
- Potenciômetro 10KΩ,
- Capacitor Cerâmico 10 nF/ 100 nF,
- Capacitor Eletrolítico 10uF/ 100uF,
- Chave Táctil (Push-Button).

Vamos revisar a função de cada um deles dentro de um circuito eletrônica e apresentar mais algumas equações fundamentais para ter em mente ao fazer suas montagens.

Resistores elétricos

Os resistores são componentes que se opõem à passagem de corrente elétrica, ou seja, oferecem uma resistência elétrica. Dessa forma, quanto maior for o valor de um resistor, menor será a corrente elétrica que fluirá por ele e pelo condutor a ele conectada. A unidade de resistência elétrica é o **Ohm(Ω)**, que é a unidade usada para especificar o valor dos resistores.

Os resistores mais comuns do mercado são construídos com fio de carbono e são vendidos em várias especificações. Os resistores do Kit são os tradicionais de ¼ W e 10% de tolerância. Isso significa que eles podem dissipar no máximo ¼ W (0,25 watts) e seu valor de resistência pode variar em até 10%. O resistor de 1 kΩ pode então ter um valor mínimo de 900 Ω e um valor máximo de 1100 Ω.

Em algumas aplicações você pode precisar de resistores com precisão maior, como 5% ou 1%. Em outros casos, pode precisar de resistores com potência maior, como ½ W ou menor, como 1/8 W. Essas variações dependem da natureza de cada circuito.

Em geral, para as aplicações típicas e montagens de prototipagem que podem ser feitos com o Kit Arduino Iniciante, os resistores tradicionais de ¼ W e 10% de tolerância são suficientes.

Outro ponto importante de se mencionar aqui é a Lei de Ohm, que relaciona as grandezas de tensão, corrente e resistência elétrica. A lei é dada por:

$$V = R \times I$$

Ou seja, se você sabe o valor de um resistor e a tensão aplicada nos seus terminais, você pode calcular a corrente elétrica que fluirá por ele. Juntamente com a equação para calcular potência elétrica, a lei de Ohm é importante para saber se os valores de corrente e potência que os resistores de seu circuito estão operando estão adequados.

Para fechar, você deve estar se perguntando, como saber o valor de resistência de um resistor? Você tem duas alternativas: Medir a resistência usando um multímetro ou determinar o valor por meio do código de cores do resistor.

Se você pegar um dos resistores do seu kit, verá que ele possui algumas faixas coloridas em seu corpo. Essas faixas são o código de cores do resistor. As duas primeiras faixas dizem os dois primeiros algarismos decimais. A terceira faixa colorida indica o multiplicador que devemos usar. E a última faixa, que fica um pouco mais afastada, indica a tolerância.

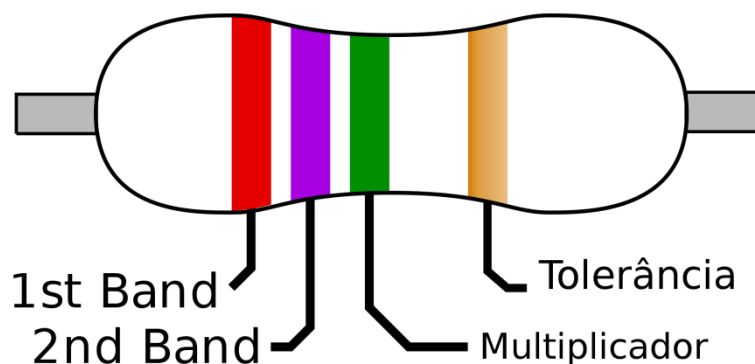



Figura 1: Faixas coloridas em um resistor

Na figura 2 apresentamos o código de cores para resistores. Cada cor está associada a um algarismo, um multiplicador e uma tolerância, conforme a tabela. Com a tabela você pode determinar a resistência de um resistor sem ter que usar o multímetro.

Mas atenção, fazer medições com o multímetro é recomendado, principalmente se o componente já tiver sido utilizado, pois ele pode ter sofrido algum dano ou mudança que não esteja visível.

Cor	Dígito	Multiplicador	Tolerância
Prata	-	x 0,01	± 10%
Dourado	-	x 0,1	± 5%
Preto	0	x 1	-
Marrom	1	x 10	± 1%
Vermelho	2	x 100	± 2%
Laranja	3	x 1K	-
Amarelo	4	x 10K	-
Verde	5	x 100K	± 0,5%
Azul	6	x 1M	± 0,25%
Violeta	7	x 10M	± 0,1%
Cinza	8	-	± 0,05%
Branco	9	-	-

Figura 2: Código de cores para resistores

Aplicando a tabela da figura 2 na imagem da figura 1, descobrimos que o resistor é de 2,7MΩ (Mega ohms) com tolerância de 5% (relativo à cor dourado da última tira).

Capacitores

Os capacitores são os elementos mais comuns nos circuitos eletrônicos depois dos resistores. São elementos que armazenam energia na forma de campos elétricos. Um capacitor é constituído de dois terminais condutores e um elemento dielétrico entre esses dois terminais, de forma que quando submetido a uma diferença de potencial, um campo elétrico surge entre esses terminais, causando o acúmulo de cargas positivas no terminal negativo e cargas negativas no terminal positivo.

São usados para implementar filtros, estabilizar sinais de tensão, na construção de fontes retificadores e várias outras aplicações. O importante que você deve saber para utilizar o Kit é que os capacitores podem ser de quatro tipos:

- Eletrolíticos,
- Cerâmicos,
- Poliéster,
- Tântalo.

Capacitor eletrolítico

As diferenças de cada tipo de capacitor são a tecnologia construtiva e o material dielétrico utilizado. Capacitores eletrolíticos são feitos de duas longas camadas de alumínio (terminais) separadas por uma camada de óxido de alumínio (dielétrico). Devido a sua construção, eles possuem **polaridade,** o que significa que você obrigatoriamente deve ligar o terminal positivo (quem tem uma “perninha” maior) no polo positivo da tensão de alimentação, e o terminal negativo (discriminado no capacitor por uma faixa com símbolos de “-”) obrigatoriamente no polo negativo da bateria. Do contrário, o capacitor será danificado.

Capacitores eletrolíticos costumam ser da ordem de micro Farad, sendo o **Farad** a unidade de medida de capacitância, usada diferenciar um capacitor do outro. A Figura 3 ilustra um típico capacitor eletrolítico.

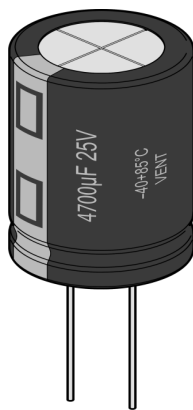



Figura 3: Capacitor eletrolítico 4700 micro Farads / 25 V

Capacitores cerâmicos

Capacitores cerâmicos não possuem polaridade, e são caracterizados por seu tamanho reduzido e por sua cor característica, um marrom claro um tanto fosco. Possuem capacitância da ordem de **pico Farad**. Veja nas imagens abaixo um típico capacitor cerâmico e sua identificação:

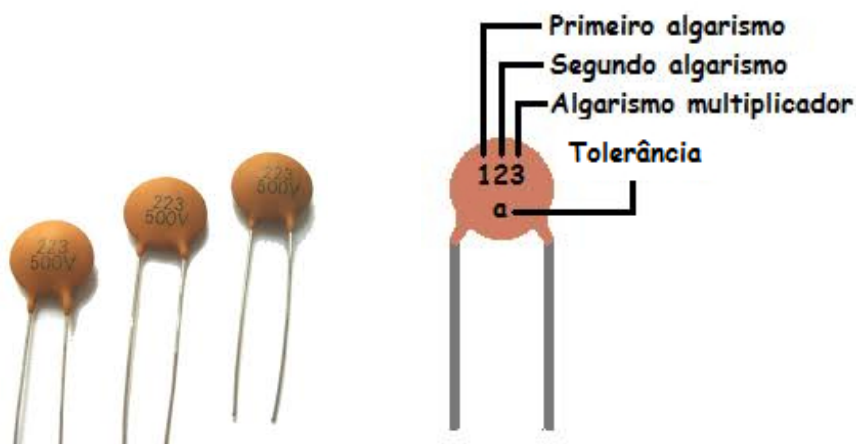



Figura 4: Capacitores cerâmicos

Na imagem da esquerda, os capacitores possuem o valor de **22 nano Farads** para a faixa de tensão de até 500V.

223 = 22 x 1000 = 22.000 pF = 22 nF

Por fim, há também os capacitores de poliéster e de tântalo. No Kit Arduino Iniciante, você receberá apenas exemplares de capacitores eletrolíticos e cerâmicos.

Diodos e Leds

Diodos e Leds são tratados ao mesmo tempo pois tratam na verdade do mesmo componente. Diodos são elementos semicondutores que só permitem a passagem de corrente elétrica em uma direção.

São constituídos de dois terminais, o **Anodo(+)** e o **catodo(-)**, sendo que para que possa conduzir corrente elétrica, é preciso conectar o Anodo na parte positiva do circuito, e o Catodo na parte negativa. Do contrário, o diodo se comporta como um circuito aberto.

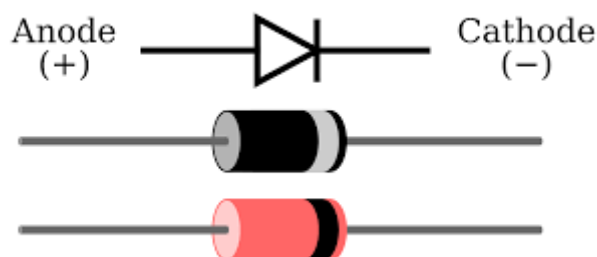



Figura 4: Diodo e seus terminais

Na figura 4, você pode ver que o componente possui uma faixa indicadora do terminal de catodo. O diodo do kit Arduino Iniciante é um modelo tradicional e que está no mercado há muitos anos, o 1N4007.

O LED é um tipo específico de diodo - **Light Emitter Diode**, ou seja, diodo emissor de luz. Trata-se de um diodo que quando polarizado corretamente, emite luz para o ambiente externo. O Kit Arduino Iniciante vem acompanhado de leds na cor vermelha, verde e amarelo, as mais tradicionais.

Nesse ponto, é importante você saber que sempre deve ligar um led junto de um resistor, para que a corrente elétrica que flua pelo led não seja excessiva e acabe por queimá-lo. Além disso, lembre-se que por ser um diodo, o led só funciona se o Anodo estiver conectado ao pólo positivo do sinal de tensão.

Para identificar o Anodo do Led, basta identificar a perna mais longa do componente, como na imagem abaixo:

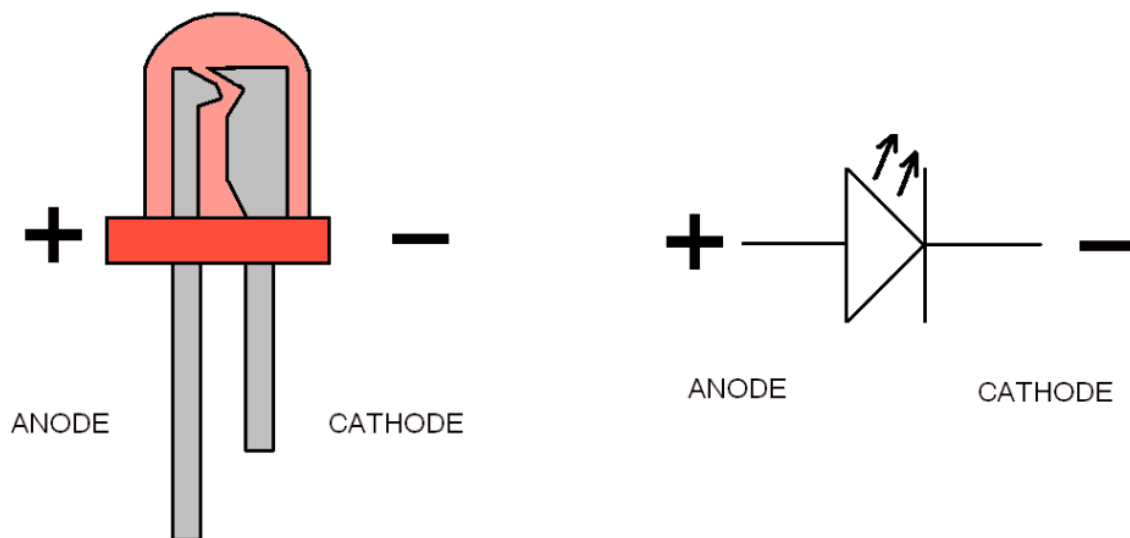



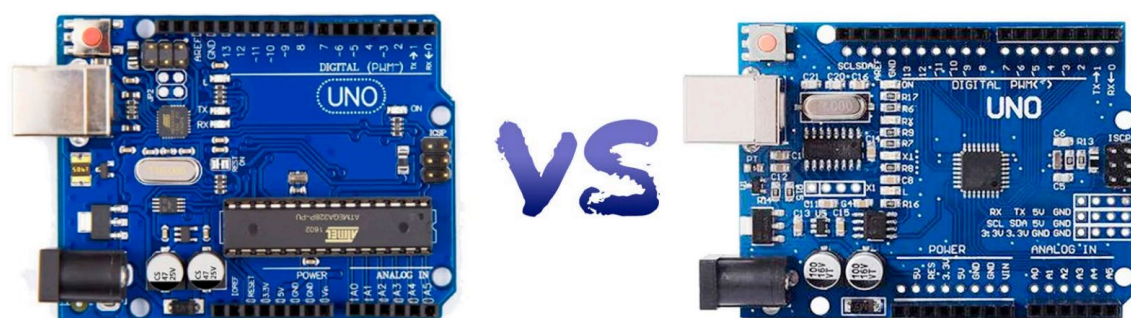
Figura 5: Terminais de um Led. Créditos: Build-eletronic-circuits.com

Os demais componentes da apostila, Buzzer, potenciômetro e push-buttons, serão explicados em seções de exemplos, nas quais iremos apresentar um circuito prático para montagem onde será explicado o funcionamento destes.

Os sensores do Kit Arduino Iniciante, quais sejam: Sensor de Luz LDR e sensor de temperatura NTC, juntamente com o Micro Servo 9g SG90 TowerPro, também terão uma seção de exemplo dedicada a cada um deles.

Parte II – Arduino Uno SMD

Após a empresa Arduino ter lançado em 2005 o Arduino Uno, essa placa se popularizou e hoje ela é a placa Arduino mais famosa do planeta e com milhões de unidades vendidas.

Algumas pessoas não sabem, mas as placas Arduino não possuem patente, e eles disponibilizam o projeto de suas placas para qualquer um reproduzir, fazer modificações e fazer melhorias. Com isso, um grupo de pessoas resolveram lançar a placa Uno SMD.

A placa Uno SMD é compatível 100% com uma placa Uno do projeto original, inclusive em todas as Shields. Algumas diferenças são meramente de projeto, como a troca de alguns tipos de componentes por outro, e a principal troca estética é a troca do chip Atmega328P no modelo PTH (que fica encaixado na placa) pelo modelo SMD.

A sigla SMD significa Surface Mounted Device, que traduzindo é algo como Dispositivo Montado em Superfície, isso significa que diferente do PTH, esse tipo de chip é soldado diretamente na superfície da placa sem precisar de furo.

Isso faz com que o chip possa ser muito menor que o PTH (que significa algo como Terminal Inserido no Furo", por não precisar fazer furos. Como pode ser visto na placa, o chip que é como o cérebro do Arduino muda demais de tamanho de uma placa para outra devido essa tecnologia.

Mas uma pergunta pode estar pairando, que é: "Mas por o chip ser menor, isso não significa que seja menos potente?". A resposta para essa pergunta é não, porque no caso do chip maior ele não tira nenhuma vantagem por ser maior, tendo dentro do chip muito espaço vazio somente para cumprir o requisito de ser PTH.

Bom, sabendo disso, agora podemos explorar outra diferença, que é o chip responsável por fazer a conversão USB-Serial. Esse chip é usado para fazer com que seu computador

reconheça sua placa Arduino, e faz a interface da “linguagem” do USB para a linguagem do Arduino, que no caso é serial UART.

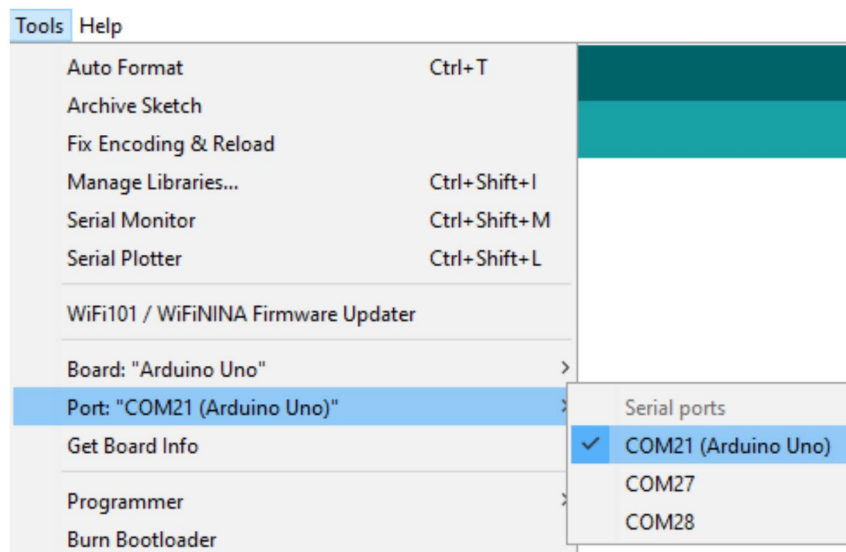
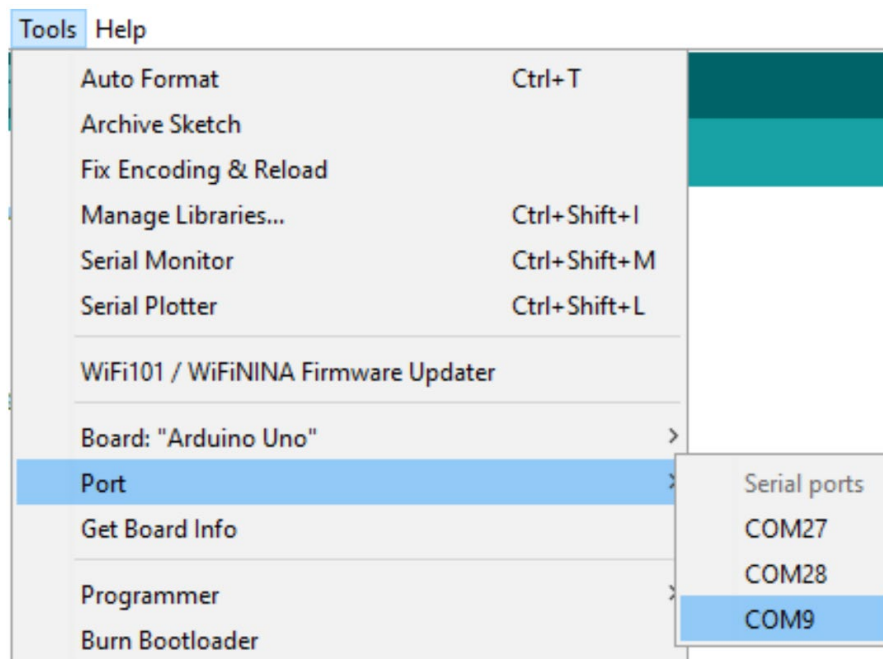
No projeto original o chip utilizado é o Atmega16U2, enquanto no modelo SMD o chip utilizado é o CH340G. No funcionamento não altera em absolutamente nada, e ambos os chips são capazes de fazer a mesma coisa, a única diferença está na instalação dos drivers e no reconhecimento automático da Arduino IDE.

O chip do Arduino original, o Atmega16U2, tem seus drivers (driver é um pequeno programa instalado no seu sistema operacional para que ele saiba como operar o dispositivo) instalados automaticamente quando você instala a sua Arduino IDE. Logo, você não necessita instalar driver nenhum para fazer funcionar sua placa.

Já na placa Uno SMD, por ela utilizar o chip CH340G, você necessita instalar o driver CH340G. Por sorte, é um processo bem simples, bastando baixar o driver e instalar no seu computador como se faz com um programa comum.

Download dos Drivers:

Windows: http://apps.eletrogate.com/CH341SER_WINDOWS.zip
Mac: http://apps.eletrogate.com/CH341SER_MAC.zip
Linux: http://apps.eletrogate.com/CH341SER_LINUX.zip

É interessante ressaltar que mesmo após as instalações dos drivers, a Arduino IDE não identifica automaticamente a placa para enviar o seu código. No caso da placa baseada no projeto original, ao plugar a placa na USB a porta COM (serial) da placa é marcada com o modelo da placa.

Enquanto no modelo SMD a placa não é marcada.

Mas isso é algo bastante simples, porque na maioria das vezes a única porta COM que aparece nesta opção é a do Arduino. Caso tenha outras, o número delas não muda, então, quando você inserir a porta nova é a do seu Arduino Uno SMD.

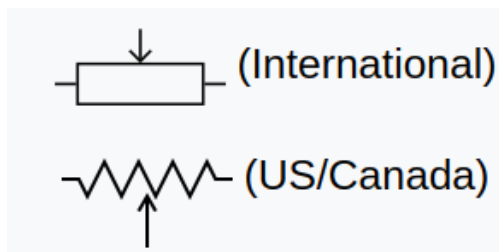
Parte III - Seção de Exemplos Práticos

Agora vamos entrar nas seções de exemplos em si. Os conceitos da Parte I são importantes caso você esteja começando a trabalhar com eletrônica. No entanto, devido ao aspecto prático da montagem, não necessariamente você precisa de ler toda a parte introdutória para montar os circuitos abaixo.

Em cada exemplo vamos apresentar os materiais utilizados, o diagrama de circuito e o código base com as devidas explicações e sugestões de referências.

Exemplo 1 - Leitura de Sinais Analógicos com Potenciômetro

O potenciômetro nada mais do que um resistor cujo valor de resistência pode ser ajustado de forma manual. Existem potenciômetros slides e giratórios. Na figura abaixo mostramos um potenciômetro giratório dos mais comumente encontrados no mercado.

Em ambos, ao variar a posição da chave manual, seja ela giratória ou slide, o valor da resistência entre o terminal de saída e um dos outros terminais se altera. O símbolo do potenciômetro é o mostrado na seguinte imagem:

Nesse exemplo, vamos ligar a saída de um potenciômetro a uma entrada analógica da Arduino UNO. Dessa forma, vamos ler o valor de tensão na saída do potenciômetro e vê-la variando de 0 a 1023. Mas como assim, 0 a 1023?

Isso se deve ao seguinte. Vamos aplicar uma tensão de 5V nos terminais do potenciômetro. A entrada analógica do Arduino, ao receber um sinal de tensão externo, faz a conversão do mesmo para um valor digital, que é representado por um número inteiro de 0 a 1023. Esses números são assim devido à quantidade de bits que o conversor analógico digital do Arduino trabalha, que são 10 bits (2 elevado a 10 = 1024).

Ou seja, o arduino divide o valor de tensão de referência em 1024 unidades (0 a 1023) de 0,00488 volts. Assim, se a tensão lida na entrada analógica for de 2,5V, o valor capturado pelo arduino será metade de 2,5/0,00488 = 512. Se for 0V, será 0, e ser for 5V, será 1023, e assim proporcionalmente para todos os valores. Assim, digamos que o valor de tensão se dado por V. O valor que o Arduino vai te mostrar é:

Valor = (V/5)*1024

Em que 5V é o valor de referência configurado no conversor analógico-digital (uma configuração já padrão, não se preocupe com ela) e 1024 é igual 2 elevado a 10. No nosso código, queremos saber o valor de tensão na saída do potenciômetro, e não esse número de 0 a 1023, assim, rearranjar a equação para o seguinte:

Tensão = Valor*(5/1024)

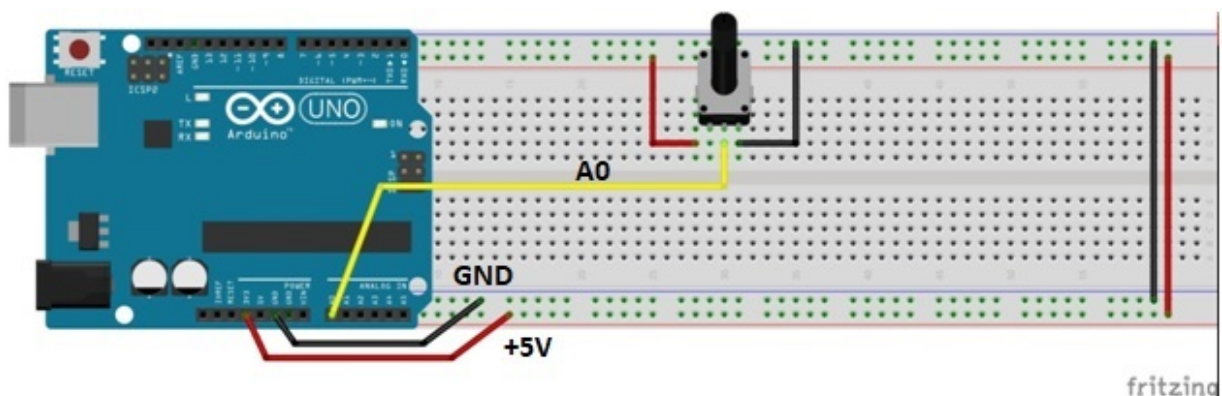
Bacana, né? Agora, vamos à montagem em si.

Lista de materiais:

Para esse exemplo você vai precisar:

- Arduino UNO;
- Protoboard;
- Potenciômetro 10K;
- Jumpers de ligação;

Diagrama de circuito

Monte o circuito conforme diagrama abaixo e carregue o código de exemplo:

O Potenciômetro possui 3 terminais, sendo que o do meio é o que possui resistência variável. A ligação consiste em ligar os dois terminais fixos a uma tensão de 5V. Assim, o terminal intermediário do potenciômetro terá um valor que varia de 0 a 5V à medida que você dá gira seu knob. O terminal intermediário é ligado diretamente a uma entrada analógica do Arduino (A0). Como a tensão é de no máximo 5V, então não há problema em ligar direto.

Carregue o código abaixo no Arduino e você verá as leituras no Monitor Serial.

```
// Exemplo 1 - Usando potenciometro para fazer leituras analógicas
// Apostila Eletrogate - KIT INICIANTE V2

#define sensorPin A0      // define entrada analógica A0

int sensorValue = 0;      // variável inteiro igual a zero
float voltage;            // variável numero fracionario

void setup()
{
  Serial.begin(9600);     // monitor serial - velocidade 9600 Bps
  delay(100);             // atraso de 100 milisegundos
}

void loop()
{
  sensorValue = analogRead(sensorPin);        // leitura da entrada analógica A0
  voltage = sensorValue * (5.0 / 1024);       // cálculo da tensão

  Serial.print("Tensão do potenciometro: "); // imprime no monitor serial
  Serial.print(voltage);                      // imprime a tensão
  Serial.print("    Valor: ");                // imprime no monitor serial
  Serial.println(sensorValue);                // imprime o valor
  delay(500);                                 // atraso de 500 milisegundos
}
```

No código, nós declaramos a variável **sensorValue** para armazenar as leituras da entrada analógica A0 e a variável do tipo **float Voltage** para armazenar o valor lido convertido para tensão.

Na função **void Setup()**, nós inicializamos o terminal serial com uma taxa de transmissão de 9600 kbps. Na função **void Loop(),** primeiro faz-se a leitura da entrada analógica A0 com a função **analogRead(SensorPin)** e armazenamos a mesma na variável **sensorValue**. Em seguida, aplicamos a fórmula para converter a leitura (que é um número entre 0 e 1023) para o valor de tensão correspondente. O resultado é armazenado na variável **Voltage** e em seguido mostrado na interface serial da IDE Arduino.

Referência:

1. https://www.arduino.cc/en/Tutorial/ReadAnalogVoltage

Exemplo 2 - Divisores de Tensão com Resistores

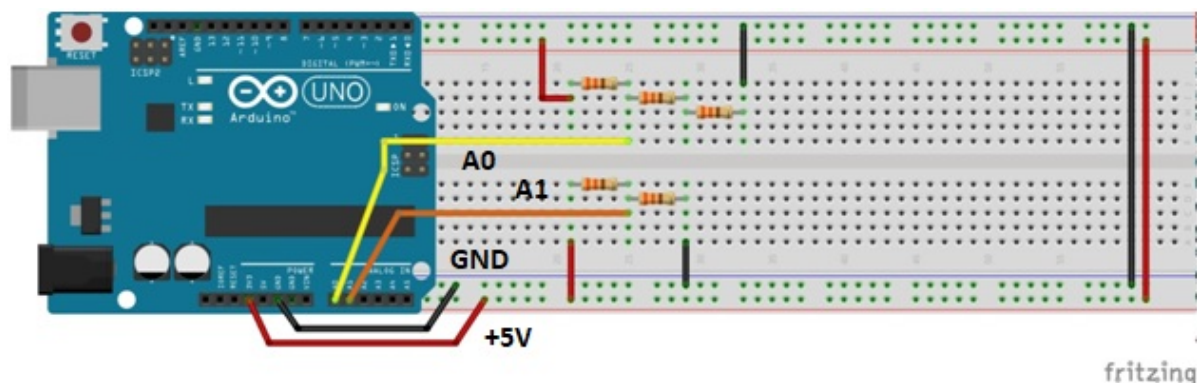
Esse exemplo é para ilustrar como usar um divisor de tensão. Sempre que você precisar abaixar um sinal lógico de 5V para 3.3V você pode usar esse circuito. Explicamos o divisor de tensão na seção de introdução, mais especificamente, quando conversamos sobre conversão de níveis lógicos.

Esse circuito será útil sempre que você tiver que abaixar as saídas do Arduino de 5V para 3.3V.

Lista de materiais:

Para esse exemplo você vai precisar:

- 3 resistores de 330 ohms;
- 2 resistores de 1K ohm;
- 1 Arduino UNO;
- Protoboard;
- Jumpers de ligação;

Diagrama de circuito:

Esse é um diagrama com dois divisores de tensão. O primeiro é composto por três resistores, sendo cada um de 330. Assim, o resistor Z1 é de 330 e o resistor Z2 é a associação série dos outros dois, resultando numa resistência de 660. Dessa forma, a tensão de saída do divisor é:

(660 / 330 + 660) * 5 = 3,33V

O segundo divisor é formado por dois resistores de 1K, dessa forma, a tensão de saída é a tensão de entrada dividida pela metade:

(1000 / 1000 + 1000) * 5 = 2,5 V

O código para o exemplo 2 é uma extensão do código usada na seção anterior para ler valores de tensão do potenciômetro. Nesse caso, nós fazemos a leitura de dois canais analógicos (A0 e A1), fazemos as conversões para as tensões e mostramos os resultados de cada divisor na interface serial.

Carregue o código abaixo e observe os dois valores no Monitor Serial da IDE Arduino.

```
// Exemplo 2 - Divisor de tensão
// Apostila Eletrogate - KIT INICIANTE V2

#define sensorPin1 A0           // define entrada analógica A0
#define sensorPin2 A1           // define entrada analógica A1

int sensorValue1 = 0;           // variavel inteiro igual a zero
int sensorValue2 = 0;           // variavel inteiro igual a zero
float voltage1;                 // variavel numero fracionario
float voltage2;                 // variavel numero fracionario

void setup()
{
  Serial.begin(9600);           // monitor serial - velocidade 9600 Bps
  delay(100);                   // atraso de 100 milisegundos
}

void loop()
{
  sensorValue1 = analogRead(sensorPin1);   // leitura da entrada analógica A0
  sensorValue2 = analogRead(sensorPin2);   // leitura da entrada analógica A1
  voltage1 = sensorValue1 * (5.0 / 1024);  // cálculo da tensão 1
  voltage2 = sensorValue2 * (5.0 / 1024);  // cálculo da tensão 2
  Serial.print("Tensao do divisor 1: ");   // imprime no monitor serial
  Serial.print(voltage1);                  // imprime a tensão 1
  Serial.print("  Tensao do divisor 2: "); // imprime no monitor serial
  Serial.println(voltage2);                // imprime a tensão 2
  delay(500);                              // atraso de 500 milisegundos
}
```

Referências:

1. https://www.arduino.cc/en/Tutorial/ReadAnalogVoltage
2. http://www.ufjf.br/fisica/files/2013/10/FIII-05-07-Divisor-de-tens%C3%A3o.pdf
3. http://www.sofisica.com.br/conteudos/Eletromagnetismo/Eletrodinamica/associacaoderesistores.php

Exemplo 3 - Controle de LED's com Botões

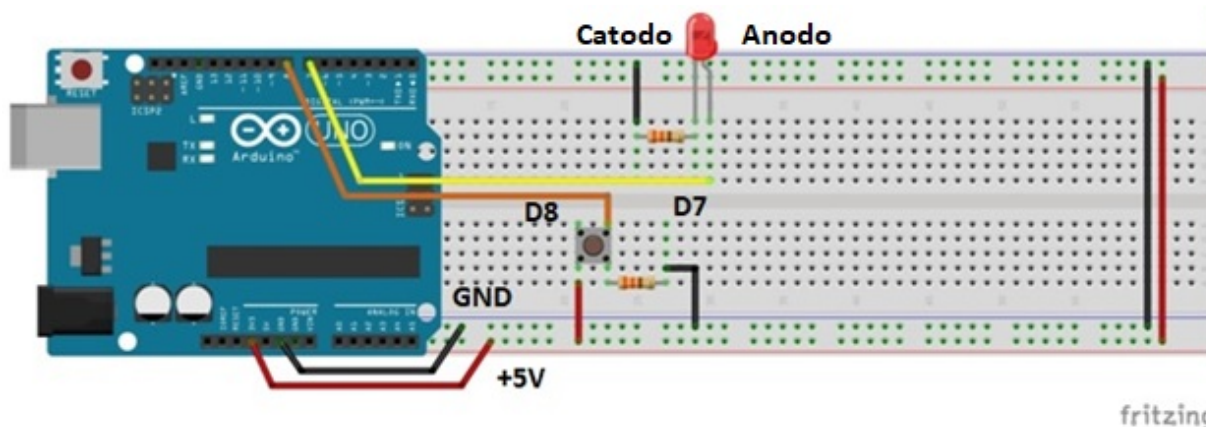
Os push-buttons (chaves botão) e leds são elementos presentes em praticamente qualquer circuito eletrônico. As chaves são usadas para enviar comandos para o Arduino e os Leds são elementos de sinalização luminosa.

Esses dois componentes são trabalhados por meio das entradas e saídas digitais do Arduino. Neste exemplo vamos fazer uma aplicação básica que você provavelmente vai repetir muitas vezes. Vamos ler o estado de um push-button e usá-la para acender ou apagar um led. Ou seja, sempre que o botão for acionado, vamos apagar ou acender o Led.

Lista de materiais:

Para esse exemplo você vai precisar:

- 2 resistores de 330 ohms;
- 1 Led vermelho (ou de outra cor de sua preferência);
- Push-button (chave botão);
- 1 Arduino UNO;
- Protoboard;
- Jumpers de ligação;

Diagrama de circuito:

O código a seguir mostra como ler entradas digitais e acionar as saídas digitais. Na função **void Setup(),** é preciso configurar qual pino será usado como saída ou entrada.

Depois de configurar os pinos, para acioná-los basta chamar a função **digitalWrite(pino,HIGH).** A função **digitalWrite()** aciona ou desarma um pino digital dependendo do valor passado no argumento. Se for "HIGH", o pino é acionado. Se for "LOW", o pino é desligado.

Na função **void Loop(),** fizemos um if no qual a função **digitalRead** é usada para saber se o pushButton está acionado ou não. Caso ele esteja acionado, nós acendemos o Led, caso ele esteja desligado, nós desligamos o led.
Carregue o código abaixo e pressione o botão para acender o LED.

```
// Exemplo 3 - Entradas e saídas digitais - push-button + led
// Apostila Eletrogate - KIT INICIANTE V2

#define PinButton 8                     // define pino digital D8
#define ledPin 7                        // define pino digital D7

void setup()
{
  pinMode(PinButton, INPUT);            // configura D8 como entrada digital
  pinMode(ledPin, OUTPUT);              // configura D7 como saída digital
  Serial.begin(9600);                   // monitor serial - velocidade 9600 Bps
  delay(100);                           // atraso de 100 milisegundos
}

void loop()
{
  if ( digitalRead(PinButton) == HIGH)   // se chave = nivel alto
  {
    digitalWrite(ledPin, HIGH);          // liga LED com 5V
    Serial.print("Acendendo Led");       // imprime no monitor serial
  }
  else                                   // senão chave = nivel baixo
  {
    digitalWrite(ledPin, LOW);           // desliga LED com 0V
    Serial.print("Desligando led");      // imprime no monitor serial
  }
  delay(100);                            // atraso de 100 milisegundos
}
```

Referências:

- https://www.arduino.cc/reference/en/language/functions/digital-io/digitalread/
- https://www.arduino.cc/reference/en/language/functions/digital-io/digitalwrite/

Exemplo 4 - Acionamento de LED conforme do Luz Ambiente

O sensor LDR é um sensor de luminosidade. LDR é um **Light Dependent Resistor**, ou seja, um resistor cuja resistência varia com a quantidade de luz que incide sobre ele. Esse é seu princípio de funcionamento.

Quanto maior a luminosidade em um ambiente, menor a resistência do LDR. Essa variação na resistência é medida através da queda de tensão no sensor, que varia proporcionalmente (de acordo com a lei Ohm, lembra?) com a queda na resistência elétrica.

A imagem abaixo mostra o sensor em mais detalhes:

Fotoresistor (LDR)

É importante considerar a potência máxima do sensor, que é de 100 mW. Ou seja, com uma tensão de operação de 5V, a corrente máxima que pode passar por ele é 20 mA. Felizmente, com 8K ohms (que medimos experimentalmente com o ambiente bem iluminado), que é a resistência mínima, a corrente ainda está longe disso, sendo 0,625mA. Dessa forma, podemos conectar o sensor diretamente com o Arduino.

**Nota:* Nas suas medições, pode ser que você encontre um valor de resistência mínimo diferente, pois depende da iluminação local.*

Especificações do LDR:

- Modelo: GL5528
- Diâmetro: 5mm
- Tensão máxima: 150VDC
- Potência máxima: 100mW
- Temperatura de operação: -30°C a 70°C
- Comprimento com terminais: 32mm
- Resistência no escuro: 1 MΩ (Lux 0)
- Resistência na luz: 10-20 KΩ (Lux 10)

Este sensor de luminosidade pode ser utilizado em projetos com arduino e outros microcontroladores para alarmes, automação residencial, sensores de presença e vários outros.

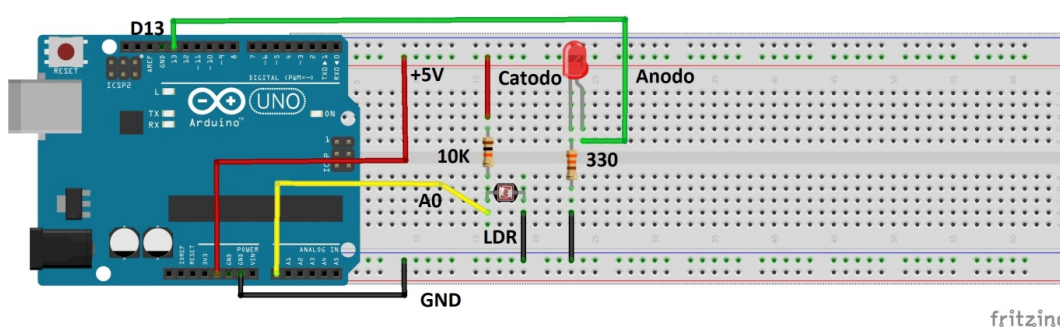
Nesse exemplo, vamos usar uma entrada analógica do Arduino para ler a variação de tensão no LDR e, consequentemente, saber como a luminosidade ambiente está se comportando. Veja na especificação que com muita luz, a resistência fica em torno de 10-20 KΩ, enquanto no escuro pode chegar a 1MΩ.

Para podermos ler as variações de tensão resultantes da variação da resistência do LDR, vamos usar o sensor como parte de um divisor de tensão. Assim, a saída do divisor será dependente apenas da resistência do sensor, pois a tensão de entrada e a outra resistência são valores conhecidos. No nosso caso, vamos usar um resistor de 10K e uma tensão de operação de 5V. Assim, o sinal que vamos ler no arduino terá uma variação de 2,2V (quando o LDR for 8K) e 5V (quando o LDR tiver resistências muito maiores que o resistor de 10K).

Lista de materiais:

Para esse exemplo você vai precisar:

- LDR;
- Resistor de 10k;
- 1 Arduino UNO;
- Protoboard;
- Jumpers de ligação;

Diagrama de circuito:

No diagrama, o sensor é ligado como parte de um divisor de tensão no pino analógico A0, de forma que a tensão de saída do divisor varia de acordo com a variação da resistência do sensor. Assim, vamos identificar as variações na intensidade de luz pelas variações na tensão do sensor. Quanto maior a intensidade de luz, menor a resistência do sensor e, consequentemente, menor a tensão de saída.

Carregue o código abaixo e varie a luminosidade sobre o LDR.

```
// Exemplo 4 - Sensor de luz LDR
// Apostila Eletrogate - KIT INICIANTE V2

#define AnalogLDR A0                  // define pino analógico A0
#define Limiar 1.5                    // define constante igual a 1.5
#define ledPin 13                     // define pino digital D13

int Leitura = 0;                      // variavel inteiro igual a zero
float VoltageLDR;                     // variavel numero fracionario
float ResLDR;                         // variavel numero fracionario

void setup()
{
  pinMode(ledPin, OUTPUT);                // configura D13 como saída digital
  Serial.begin(9600);                     // monitor serial - velocidade 9600 Bps
  delay(100);                             // atraso de 100 milisegundos
}

void loop()
{
  Leitura = analogRead(AnalogLDR);         // leitura da tensão no pino analogico
A0
  VoltageLDR =  Leitura * (5.0/1024);     // calculo da tensão no LDR
  Serial.print("Leitura sensor LDR = ");  // imprime no monitor serial
  Serial.println(VoltageLDR);             // imprime a tensão do LDR

  if (VoltageLDR > Limiar)                // se a tensão LDR maior do que limiar
    digitalWrite(ledPin, HIGH);           // liga LED com 5V
  else                                    // senão a tensão LDR < limiar
    digitalWrite(ledPin, LOW);            // desliga LED com 0V
  delay(500);                             // atraso de 500 milisegundos
}
```

No código acima, usamos funcionalidades de todos os exemplos anteriores. Como o sensor é um elemento de um divisor de tensão, fazemos a sua leitura do mesmo modo que nos exemplos do potenciômetro e divisor de tensão. Nesse caso, definimos uma tensão de limiar, a partir da qual desligamos ou ligamos um led para indicar que a intensidade da luz ultrapassou determinado valor. Esse limiar pode ser ajustado por você para desligar o led em intensidades diferentes de luz ambiente.

É importante que o sensor esteja exposto à iluminação ambiente e não sofra interferência de fontes luminosas próximas, mas que não sejam parte do ambiente.

Referências:

- https://maker.pro/education/using-an-ldr-sensor-with-arduino-a-tutorial-for-beginners
- http://blog.eletrogate.com/controle-de-luminosidade-com-arduino-e-sensor-ldr/

Exemplo 5 - Medindo a Temperatura do Ambiente

O sensor de temperatura NTC pertence a uma classe de sensores chamada de Termistores. São componentes cuja resistência é dependente da temperatura. Para cada valor de temperatura absoluta há um valor de resistência. Assim, o princípio para medir o sinal de um termistor é o mesmo que usamos para medir o sinal do LDR. Vamos medir na verdade, a queda de tensão provocada na resistência do sensor.

Existem dois tipos básicos de termistores:

- **PTC (Positive temperature coeficient):** Nesse sensor, a resistência elétrica aumenta à medida que a temperatura aumenta;
- **NTC (Negative Temperature Coeficient):** Nesse sensor, a resistência elétrica diminui à medida que a temperatura aumenta.

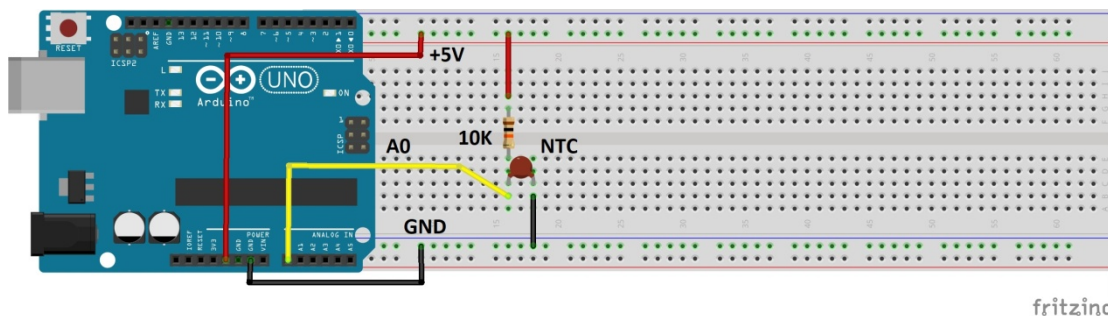
O termistor que acompanha o kit é do tipo NTC, como o próprio nome diz. Esse é o tipo mais comum do mercado.

O grande problema dos termistores é que é necessária fazer uma função de calibração, pois a relação entre temperatura e resistência elétrica não é linear. Existe uma equação, chamada equação de **Steinhart-Hart** que é usada para descrever essa relação (vide referências). O código que vamos usar nesse exemplo utiliza a equação de Steinhart-Hart conforme a referência 1.

Lista de materiais:

Antes de mais nada, vamos separar os componentes que precisamos para montar o NTC junto com o arduino. A nossa lista de componentes é a seguinte:

- 1 Sensor de temperatura NTC;
- Arduino UNO + cabo USB;
- 1 resistor de 10k;
- Protoboard;
- Jumpers para conexão no protoboard;

Diagrama de circuito:

Para uma melhor precisão nas medidas de temperatura, meça a tensão de 5V no barramento do protoboard, usando um multímetro (não contém no KIT). Essa tensão é

a mesma usada como referência do conversor analógico-digital do Arduino. Especifique esse valor de tensão na primeira linha do Sketch. Por exemplo, se a tensão for 4,85 V:
#define Vin 4.85

Carregue o código abaixo e meça a temperatura com o NTC:

```
// Exemplo 5 - Sensor de temperatura NTC
// Apostila Eletrogate - KIT INICIANTE V2

#define Vin 5.0                    // define constante igual a 5.0
#define T0 298.15                  // define constante igual a 298.15 Kelvin
#define Rt 10000                   // Resistor do divisor de tensao
#define R0 10000                   // Valor da resistencia inicial do NTC
#define T1 273.15                  // [K] in datasheet 0º C
#define T2 373.15                  // [K] in datasheet 100° C
#define RT1 35563                  // [ohms] resistencia in T1
#define RT2 549                    // [ohms] resistencia in T2
float beta = 0.0;                  // parametros iniciais [K]
float Rinf = 0.0;                  // parametros iniciais [ohm]
float TempKelvin = 0.0;            // variable output
float TempCelsius = 0.0;           // variable output
float Vout = 0.0;                  // Vout in A0
float Rout = 0.0;                  // Rout in A0

void setup()
{
  Serial.begin(9600);                         // monitor serial - velocidade 9600 Bps
  beta = (log(RT1 / RT2)) / ((1 / T1) - (1 / T2));     // calculo de beta
  Rinf = R0 * exp(-beta / T0);                // calculo de Rinf
  delay(100);                                 // atraso de 100 milisegundos
}

void loop()
{
  Vout = Vin * ((float)(analogRead(0)) / 1024.0); // calculo de V0 e leitura de A0
  Rout = (Rt * Vout / (Vin - Vout));              // calculo de Rout
  TempKelvin = (beta / log(Rout / Rinf));         // calculo da temp. em Kelvins
  TempCelsius = TempKelvin - 273.15;              // calculo da temp. em Celsius
  Serial.print("Temperatura em Celsius: ");       // imprime no monitor serial
  Serial.print(TempCelsius);                      // imprime temperatura Celsius
  Serial.print("   Temperatura em Kelvin: ");     // imprime no monitor serial
  Serial.println(TempKelvin);                     // imprime temperatura Kelvins
  delay(500);                                     // atraso de 500 milisegundos
}
```

Referências e sugestões de leitura:

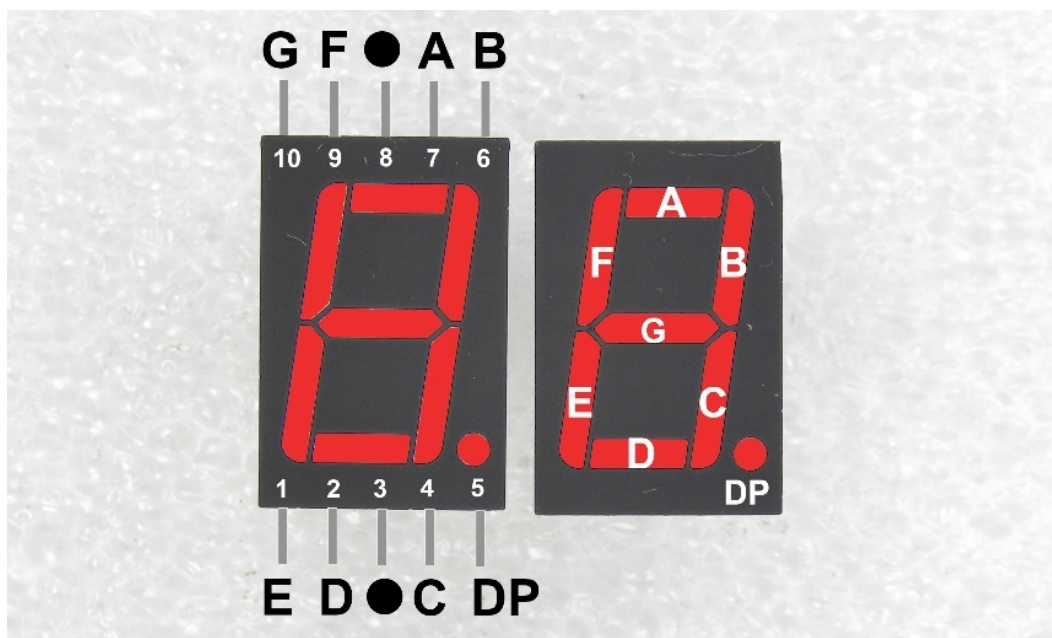
1. http://www.instructables.com/id/NTC-Temperature-Sensor-With-Arduino/
2. https://www.ametherm.com/thermistor/ntc-thermistors-steinhart-and-hart-equation
3. http://www.thinksrs.com/downloads/PDFs/ApplicationNotes/LDC%20Note%204%20NTC%20Calculator.pdf
4. https://www.mundodaeletrica.com.br/sensor-de-temperatura-ntc-ptc/

Exemplo 6 – Contador com Display de 7 Segmentos

Os displays de 7 segmentos foram desenvolvidos a partir da evolução dos Leds. Como já sabe, o Led é um diodo de material semicondutor que quando a corrente elétrica é passada por ele, uma luz é emitida. Devido à construção desse semicondutor, é necessária uma limitação de corrente, pois senão ele queima. Por isso, para a ligação de um Led, um resistor em série deve ser conectado.

Perceba que cada segmento do Display é formado por um led. Ao acionar os segmentos, pode-se formar números e letras (com algumas limitações). É esse o princípio simples de funcionamento de um Display. Os segmentos são identificados com letras de A até G e mais DP para o ponto decimal.

Existem dois tipos de displays, com o catodo comum e com o anodo comum. Como o nome já diz, no primeiro tipo, os catodos de todos os segmentos estão conectados em um único pino negativo. E no segundo tipo, os anodos estão conectados também em um só pino, mas positivo. Portanto as polarizações dos segmentos são inversas uma da outra.

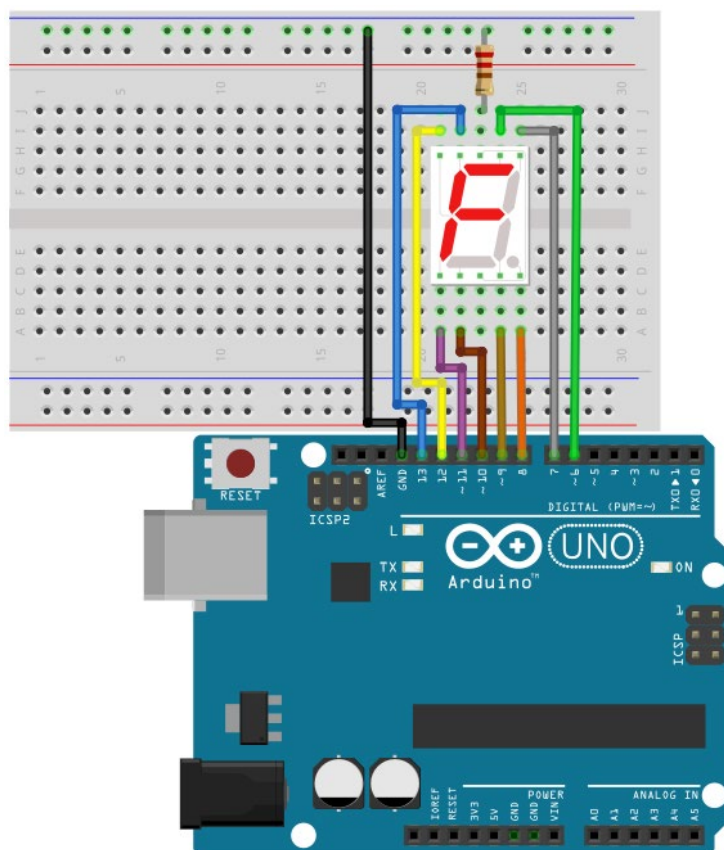
Esse é o display de 7 segmentos – anodo comum (F5161BH) utilizado nessa montagem. Os pinos com bolinhas pretas são os pinos comuns – anodos. Use somente um, já que os dois são interligados internamente. Cada segmento é identificado com uma letra – A até G. E o ponto decimal, não usado nessa montagem, é o DP.

Montagem Arduino com Display 7 segmentos

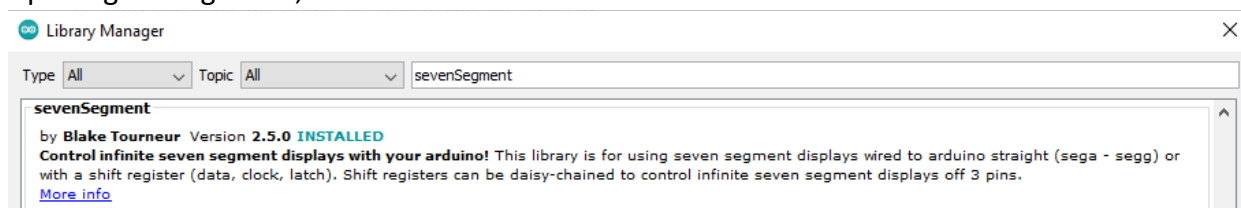
Para realizar a montagem iremos conectar o display ao Arduino utilizando os pinos de 6 a 13. O display que acompanha o kit é de cátodo comum, e isso significa que todos os LEDs estão unidos pelo ponto que vai ligado ao GND. Por isso, muita atenção, e coloque o resistor ligado ao GND.

Lista de Materiais:

Para este exemplo você vai usar os seguintes componentes:

- Arduino UNO,
- Protoboard,
- Jumpers de ligação,
- Resistor 330 ohms,
- Display 7 segmentos – anodo comum.

Diagrama do Display:

Código Arduino - Display 7 segmentos:

A biblioteca utilizada para controlar o display será a **sevenSegment**. Para instalar a nova Biblioteca na Arduino IDE, clique em: **Sketch > Incluir Biblioteca > Gerenciar Bibliotecas** Após abrir a janela do Gerenciador de Biblioteca, refine a busca digitando **sevenSegment**. Na biblioteca **sevenSegment** (de Blake Tourneur), clique em **Instalar.** Após alguns segundos, ela será automaticamente instalada.

Código Arduino:

Carregue o código presente na próxima página para a sua placa Arduino Uno. Caso o seu display seja de Anodo comum, não se preocupe, basta trocar o jumper que vai para o GND e colocar ele no 5V, e no código você altera a 4ª linha de código onde está true, você coloca false.

```
// Exemplo 6 – Display de 7 segmentos
// Apostila Eletrogate - KIT INICIANTE V2

#include  <sevenSegment.h>       // incluindo a biblioteca para que possamos usar

sevenSegment sevseg(6, 7, 9, 10, 11, 12, 13, 8, true); // Configuração dos pinos utilizados.
// Caso o seu display seja do tipo anodo comum, mude para o 'true' para 'false'

int delayTime = 1000;            // Configuração do tempo de transição do número.

void setup()
{
  sevseg.clear();     //Limpa o numero que está aparecendo e deixa o display sem nada.
}

void loop()
{
  // O código muda os números de 0 até 9, esperando 1 segundo para cada mudança

  sevseg.display('0');  // Mostra o número 0
  delay(delayTime);     // Espera o tempo que foi configurado na variável DelayTime.
  sevseg.display('1');
  delay(delayTime);
  sevseg.display('2');
  delay(delayTime);
  sevseg.display('3');
  delay(delayTime);
  sevseg.display('4');
  delay(delayTime);
  sevseg.display('5');
  delay(delayTime);
  sevseg.display('6');
  delay(delayTime);
  sevseg.display('7');
  delay(delayTime);
  sevseg.display('8');
  delay(delayTime);
  sevseg.display('9');
  delay(delayTime);
  sevseg.display('0');
  delay(delayTime);
}
```

Referências :

- http://blog.eletrogate.com/guia-completo-dos-displays-de-7-segmentos-arduino/
- https://www.nutsvolts.com/magazine/article/using-seven-segment-displays-part-1
- https://www.nutsvolts.com/magazine/article/using-seven-segment-displays-part-2

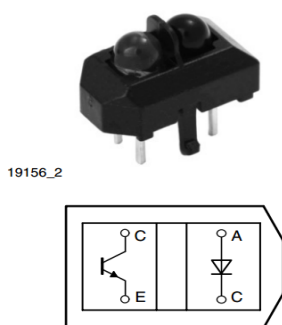
Exemplo 7 - Acendendo um LED com Sensor Reflexivo Infravermelho

O sensor ótico reflexivo **TCRT5000** é um dos mais populares para utilização em projetos com Arduino. O sensor é fabricado pela Vishay, uma tradicional fabricante de componentes eletrônicos. Recomendamos fortemente, a leitura do datasheet do sensor:

https://www.vishay.com/docs/83760/tcrt5000.pdf

Trata-se de um sensor reflexivo que possui um emissor infravermelho e um fototransistor. O emissor é um led infravermelho que emite um sinal nessa faixa do espectro. Já o fototransistor é o receptor que faz a leitura do sinal refletido. Ou seja, o led emite um feixe infravermelho que pode ou não ser refletido por um objeto. Caso o feixe seja refletido, o fototransistor identifica o sinal refletido e gera um pulso em sua saída.

Tensão direta do LED emissor é de 1,25 V com uma corrente máxima de 60 mA. A corrente máxima do fototransistor é de 100 mA.

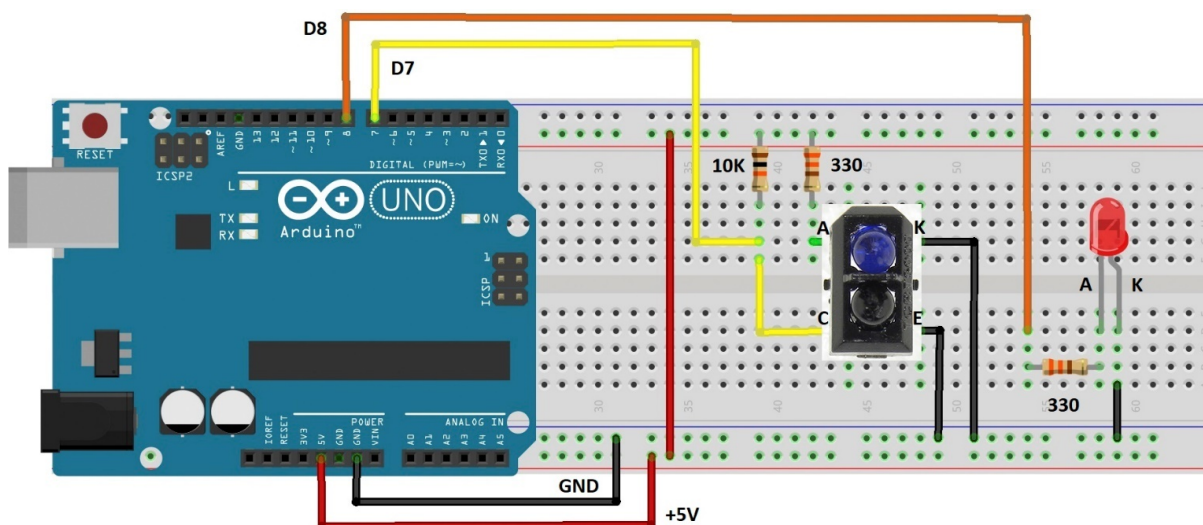
A distância máxima de detecção não é grande, ficando em torno de 25 mm (dois centímetros e meio), o que pode limitar um pouco a sua aplicação. O fototransistor vem com um filtro de luz ambiente, o que maximiza a identificação do feixe infravermelho refletido.

Sensor TCRT5000 visto por cima Fonte: Vishay

Lista de materiais:

- Arduino UNO R3;
- Protoboard;
- Sensor TCRT5000;
- Led vermelho 5mm;
- Resistor de 330;
- Resistor de 10K;
- Jumpers para ligação no protoboard;

Diagrama de Circuito:

O terminal Anodo (A) positivo do LED infravermelho está ligado por meio de um resistor de 330R ao VCC (5V) e o seu terminal catodo (K) negativo está ligado em GND. O Coletor (C) do fototransistor está ligado ao resistor de 10K (que está conectado no VCC) e também na entrada digital D7 do Arduino. Essa entrada é que vamos usar para identificar se o sensor percebeu algum objeto ou não. O Emissor (E) do fototransistor está ligado ao GND.

O LED vermelho para indicação do Sensor, tem o terminal Anodo (A) positivo conectado à um resistor de 330R que está conectado à porta digital D8 do Arduino. O terminal catodo (K) negativo está ligado ao GND.

Código:

```
// Exemplo 7 - Sensor Ótico Reflexivo TCRT5000
// Apostila Eletrogate - KIT INICIANTE V2

int leituraSensor;                  // variavel de leitura do sensor
int led = 8;                        // led indicador = D8 do Arduino
int fotoTransistor = 7;             // coletor do fototransistor = D7 do Arduino

void setup()
{
  pinMode(led, OUTPUT);             // pino do led indicador = saida
  pinMode(fotoTransistor, INPUT);   // pino do coletor do fototransistor = entrada
}

void loop()
{
  leituraSensor = digitalRead(fotoTransistor);  // leitura do sensor TCRT5000
  if (leituraSensor == 0 )                      // se o led refletir a luz
    digitalWrite(led, HIGH);                    // acende LED indicador
  else                                          // senão
    digitalWrite(led, LOW);                     // apaga LED indicador
  delay(500);                                   // atraso de 0,5 segundos
}
```

O código para utilização do sensor TCRT5000 é bem simples. Com o circuito montado, basta monitorar o estado do pino no qual a saída do sensor (coletor do fototransistor) está ligada. Se esse pino estiver em nível baixo, é porque algum objeto está presente dentro do raio de medição do sensor (a luz infra-vermelha do LED está sendo refletida). Se o pino estiver em nível alto, então não há objeto nenhum no raio de medição do sensor.

Nesse exemplo, a presença do sensor é identificada pelo acionamento de um led, ou seja, sempre que um objeto for identificado pelo sensor, o led ficará aceso.

Referências:

- http://blog.eletrogate.com/sensor-optico-tcrt5000-com-arduino/
- http://www.instructables.com/id/Using-IR-Sensor-TCRT-5000-With-Arduino-and-Program

Considerações finais

Essa apostila tem por objetivo apresentar alguns exemplos básicos sobre como utilizar os componentes do Kit Arduino Iniciante, a partir dos quais você pode combinar e fazer projetos mais elaborados por sua própria conta.

Nas seções de referências de cada exemplo e nas referências finais, também tentamos indicar boas fontes de conteúdo objetivo e com projetos interessantes. Sobre isso ponto, que consideramos fundamental, gostaríamos de destacar algumas fontes de conhecimento que se destacam por sua qualidade.

O fórum oficial Arduino possui muitas discussões e exemplos muito bons. A comunidade de desenvolvedores é bastante ativa e certamente pode te ajudar em seus projetos. No Project Hub poderá encontrar milhares de projetos com Arduino.

- Fórum oficial Arduino: https://forum.arduino.cc/
- Project Hub Arduino : https://create.arduino.cc/projecthub

O Instructables é a ótima referência do mundo maker atual. Pessoas que buscam construir suas próprias coisas e projetos encontram referências e compartilham suas experiências no site.

- Instructables: https://www.instructables.com/

O Maker pro é outro site referência no mundo em relação aos projetos com Arduino. Há uma infinidade de projetos, todos bem explicados e com bom conteúdo.

- Maker pro: https://maker.pro/projects/arduino

Em relação à eletrônica, teoria de circuitos e componentes eletrônicos em geral, deixamos alguns livros essenciais na seção finais de referências. O leitor que quer se aprofundar no mundo da eletrônica certamente precisará de um livro basilar e de bons conhecimentos em circuitos eletro-eletrônicos.

No mais, esperamos que essa apostila seja apenas o início de vários outros projetos e, quem sabe, a adoção de Kits mais avançados, como os de **Robótica** e **Arduino Advanced**. Qualquer dúvida, sugestão, correção ou crítica a esse material, fique à vontade para relatar em nosso blog oficial: http://blog.eletrogate.com/

Referências gerais

1. Fundamentos de Circuitos Elétricos. Charles K. Alexander; Matthew N. O. Sadiku. Editora McGraw-Hill.
2. Circuitos elétricos. James W. Nilsson, Susan A. Riedel. Editora: Pearson; Edição: 8.
3. Microeletrônica - 5ª Ed. - Volume Único (Cód: 1970232). Sedra,Adel S. Editora Pearson.
4. Fundamentals of Microelectronics. Behzad Razavi. Editora John Wiley & Sons; Edição: 2nd Edition (24 de dezembro de 2012).

Abraços e até a próxima!

Autores:

Vitor Vidal
Engenheiro eletricista, mestrando em eng. elétrica e apaixonado por eletrônica, literatura, tecnologia e ciência. Divide o tempo entre pesquisas na área de sistemas de controle, desenvolvimento de projetos eletrônicos e sua estante de livros.

Gustavo Murta
Técnico em eletrônica, formado em Curso superior de TPD, pós-graduado em Marketing. Trabalhou por muitos anos na IBM na área de manutenção de computadores de grande porte. Aposentou-se, podendo curtir o que mais gosta : estudar e ensinar Tecnologia. Hobbista em eletrônica desde 1976. Gosta muito de Fotografia e Observação de aves.

Gustavo Nery
Graduado em Engenharia de Controle e Automação pela UFMG. Apaixonado por eletrônica, computação e tecnologias na área de sistemas embarcados. Nos tempos livres me divido entre desenvolver pesquisa na universidade, adquirir novos conhecimentos e estar com a família.

APOSTILA KIT

ARDUINO INICIANTE

Esta apostila acompanha o **Kit ARDUINO INICIANTE** da Eletrogate, e contém conteúdos relacionados a todos os componentes do Kit.

WWW.ELETROGATE.COM